

expiration date is also saved in list 130. Public key list 130 may include any number of public keys and associated expiration dates. However, when new key pairs are generated periodically for environment 10 and each key pair is set to expire on an annual basis, list 130 includes, typically, around 12-14 records. This low number of records consumes a minimal amount of memory, even when duplicated at all the authentication nodes in network 18 and helps in keeping the expense and complexity of the authentication system of the present invention low.

A task 132 is performed when a message is received that carries a record from the user data base 66 (see FIG. 7). Such messages result from the performance of task 74 (FIG. 5) by authentication center 30. In the preferred embodiment of the present invention a user data base 66, similar to the one at authentication center 30, is maintained at the authentication node. User data base 66 stored at the authentication node contains records only for those users that are clients of the node, recipients of a service. These client users may be local or visitors. Local user client records are sent directly to the authentication node from authentication center 30 by task 74 (see FIG. 5). Visitor user client records are retrieved from the authentication node to which they are local user clients on an as needed basis. Each authentication node maintains a user database 66 in a current form by receiving messages from time to time.

FIG. 11 shows a flow chart of procedure 127, which is performed by an authentication node when a log-on message 116 (see FIG. 8) is received from a user terminal 12. With reference to FIGS. 8 and 11, message 116 is received during a task 136. The expiration date is extracted in a task 138, and a task 140 uses this expiration date to perform, for example, a table look-up operation. The look-up operation utilizes public key list 130 (see FIG. 12) to select the public key that is associated with the received expiration date. Of course, if an invalid expiration date is retrieved an appropriate message may be returned to user terminal 12 (not shown). Once this public key has been selected, a task 142 decrypts encrypted block 94 from message 116. The preferred embodiment utilizes a conventional decryption process, for example, such as the RSA process. This decryption returns the data included in encrypted block 94 to its original and useful form.

After task 142, a task 144 evaluates the EDC portion of block 94, now decrypted. As is conventional, task 144 processes the user ID and equipment ID from block 94 to detect some correspondence between that data and the EDC. Well known parity, checksum, and CRC processes, for example, are contemplated at task 144. A query task 146 switches program control in response to this evaluation. A lack of such correspondence indicates that an error has occurred. In this situation, an error signifies that encrypted block 94 has not decrypted correctly. Incorrect decryption is one indicator of tampering with AM 46 (see FIGS. 2-3). It is also an indicator of a potential pirate trying to gain access to network 18 by transmitting "guesses" about acceptable encrypted blocks 94. When an error is indicated, a task 148 constructs and sends the "Deny Access" message back to user terminal 12. No further resources of network 18 are wasted on dealing with the request for service presented by message 116. User terminal 12 will act upon the "Deny Access" message as discussed above in connection with task 124 (see FIG. 9).

On the other hand, if task 146 determines that block 94 decrypted correctly, a task 150 extracts the authentication equipment ID from block 94 and the identifying equipment ID from block 112. A task 152 evaluates these two equipment ID values for some sort of correspondence. In the preferred embodiment of the present invention, these two values are simply compared for equality. However, those skilled in the art will appreciate that additional processing of one or more of these two values may take place to determine if the two values are related to one another in a predetermined manner. A query task 154 switches program control in response to this evaluation. A lack of correspondence indicates that an AM 46 does not match the user terminal 12 with which it is being used. When task 154 detects a lack of correspondence, task 148 is performed to deny access. Thus, a potential pirate does not profit by stealing an AM 46 for use in another user terminal 12 or by copying an AM 46 for use in another terminal 12.

If task 154 determines that the two values correspond, then task 155 evaluates the sequence number portion of block 94 (see FIG. 8). For every new log-on message 116 the authentication node receives from a particular terminal 12, the sequence number portion of block 94 must be greater than every sequence number contained in every previous log-on messages from the same terminal 12. Task 155 checks to see if this is true by comparing the sequence number portion of block 94 with the call count 75 field of the user data base 66 (FIG. 7). The call count 75 is initialized when the authentication nodes receives a message from the authentication center through support procedure 126 during task 132 (see FIG. 10). When the sequence number is less than or equal to call count 75, task 155 switches program control to task 148. A "Deny Access" message is sent and no further resources of network 18 are used to process this log-on message. When the sequence number is greater than the call count 75, task 156 replaces the value in the call count 75 in the user data base 66 with the value in the sequence number portion of block 94 and passes control to task 162 which constructs and sends an "Access Granted" message to terminal 12. At this point authentication is complete and the authentication node continues with other operations. The user terminal 12 will process the Access Granted message as discussed above in connection with FIG. 9. The user will be able to utilize the services and resources offered by network 18.

In summary, the present invention provides an improved system and method for authenticating, to a high level of confidence, users of a service. The high level of confidence is achieved, at least in part, by the use encrypted authentication blocks which are refreshed at regular intervals. In accordance with the present invention, the service provider is not required to control the manufacture, distribution, or use of user terminals. Rather, security is maintained through the production of inexpensive authentication modules. Moreover, in accordance with the present invention, legitimate service subscribers are not burdened by authentication procedures or systems. Once authentication modules are installed in user terminals, the terminals are available for use without the user needing to take further action with respect to authentication.

Furthermore, the cost of authentication in accordance with the present invention is minimal. The authentication modules are inexpensive. Procedures performed at user terminals and authentication nodes re-